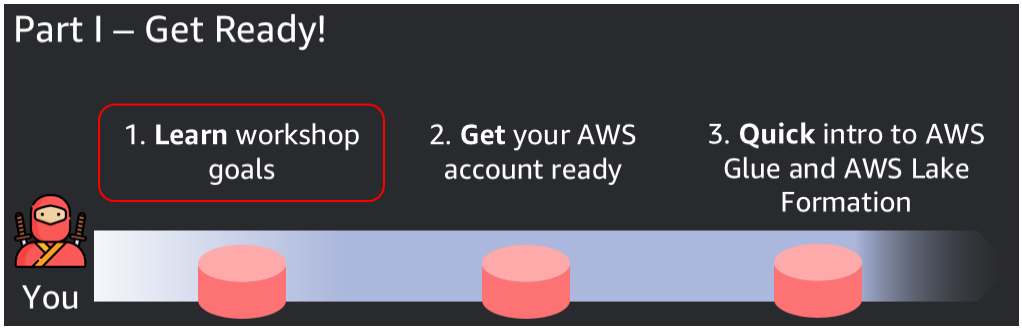
ANT201

Building Workflows on AWS Lake Formation & AWS Glue

Workshop Lab Guide

TABLE OF CONTENTS

I.1 LEARN WORKSHOP GOALS



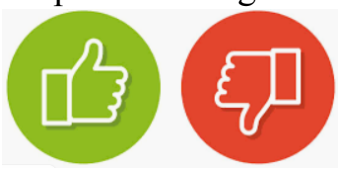
You are a data engineer at **ABCWidget**. You've been asked to help with the following tasks.

- Ingest data from a database into an S3 based data lake
- Explore the raw data
- Build an ETL pipeline and transform that data into an optimized format for faster data analytics
- Apply fine grain access security controls on the dataset and grant data analysts the ability to query the data

Amazon reviews data set
https://registry.opendata.aws/amazon-reviews

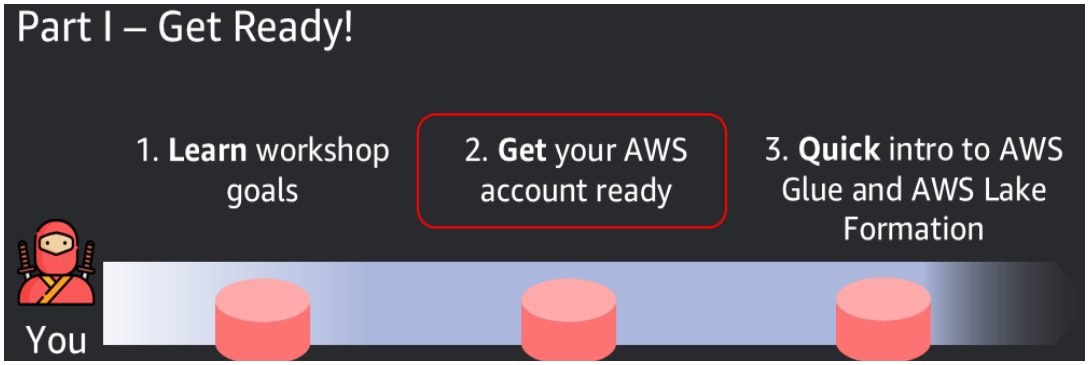
	Column name	Data type
1	marketplace	string
2	customer_id	string
3	review_id	string
4	product_id	string
5	product_parent	string
6	product_title	string
7	star_rating	int
8	helpful_votes	int
9	total_votes	int
10	vine	string
11	verified_purchase	string
12	review_headline	string
13	review_body	string
14	review_date	date
15	year	int
16	product_category	string

160M+ customer reviews across close to 2M products across numerous product categories



For the sake of brevity and speed, we have chosen to work with a subset of the Amazon reviews dataset in this workshop. The data set contains 2.4 million reviews from ‘Grocery’ product category.

I.2 GET YOUR AWS ACCOUNT READY



NOTE: If you have received AWS credentials to use for the workshop, then we have already created and setup an AWS account for you. Please follow the instructions in this side note to retrieve your Amazon S3 bucket name.

1. Navigate to **console.aws.amazon.com**
2. Sign into your AWS account with your credentials
3. From the top-right menu, change region to **Ireland (eu-west-1)**
4. Navigate to the **AWS CloudFormation** console
5. Click on the stack named **mod-XXXXX**
6. Expand the **Outputs** tab, then copy all the values into a notepad.

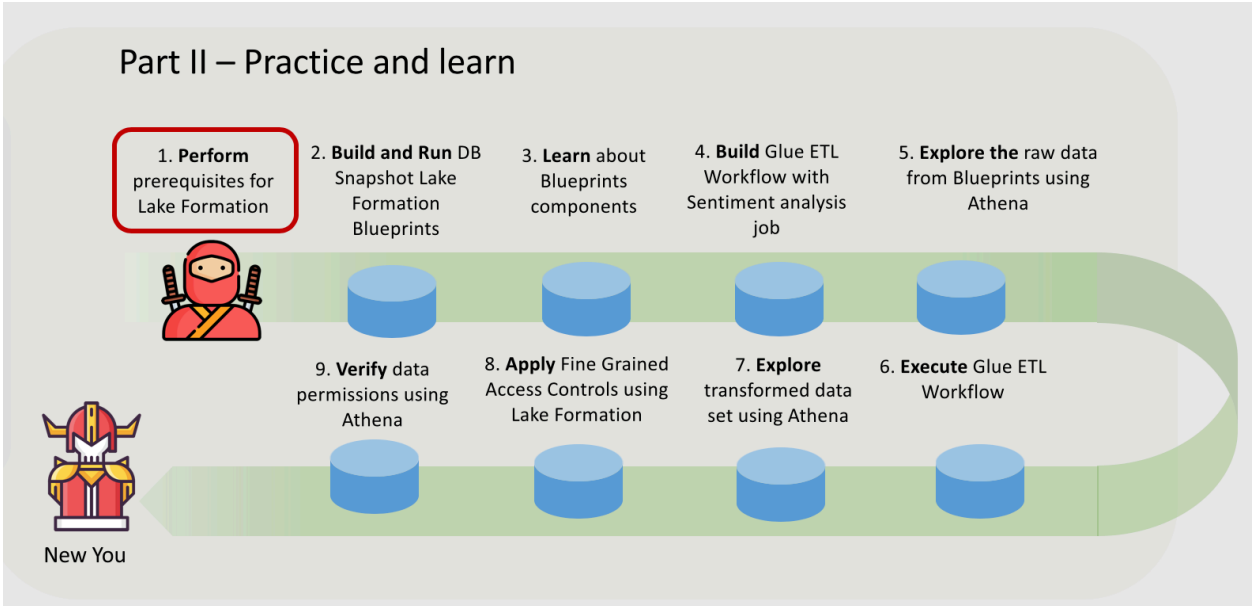
   Keep it in an open text file for reference in later exercises.

Once done, skip the remainder of the "Get your AWS account ready" section.

The AWS CloudFormation template will create the following key resources for you:

- MySQL RDS database with Amazon grocery reviews data
- S3 bucket which would serve as a Data Lake
- Glue JDBC Connection to the source MySQL database
- Glue catalog database
- Glue jobs and crawlers to build a Glue workflow for ETL
- Saved Athena queries to explore both raw and transformed data
- Necessary IAM users and roles

II.1 PERFORM PREREQUISTIES TO SETUP LAKE FORMATION



**** You are already logged in as Super Admin ****

II.1.1 CREATE A DATA LAKE ADMINISTRATOR

One of the first steps getting started with Lake Formation is adding the data lake administrator. In this step we will designate an existing AWS Identity and Access Management (IAM) user as the data lake administrator in AWS Lake Formation.

1. You are already logged in as **Super administrator (Team Role)**.
2. Navigate to **Lake formation** console.
3. Do one of the following:
    1. If a welcome message appears, choose **Add administrators**.
    2. In the navigation pane, choose **Admins and database creators**. Then under **Data lake administrators**, choose **Grant**.
4. In the **Manage data lake administrators** dialog box, add the user datalake_admin. Then choose **Save**.

II.1.2 CHANGE DATA CATALOG SETTINGS

Lake Formation starts with the "Use only IAM access control" settings enabled for compatibility with existing AWS Glue Data Catalog behavior. We recommend that you disable these settings after you transition to using Lake Formation permissions.

1. Ensure that you are still signed in to the Lake Formation console **as the Super administrator user.**
2. In the navigation pane, under **Data catalog**, choose **Settings**.
3. **Uncheck** both **check boxes** and choose **Save**.
4. Click on the dropdown menu on the top right where it indicates the logged in user and copy the account number to a notepad. Remove the dashes from the account number. Sign out of AWS console and sign back in as the data lake administrator, the user **datalake_admin**. Sign in using the **account number** that you copied to notepad, **datalake_admin** as username and **Welcome1** as password.

   **** GOING FORWARD YOU SHOULD BE LOGGED IN AS datalake_admin USER ****

5. Navigate back to Lake Formation console and in the navigation pane on the left, under **Permissions**, choose **Admins and database creators**.
6. Under **Database creators**, select the **IAMAllowedPrincipals group**, and choose **Revoke**.

   The **Revoke** permissions dialog box appears, showing that IAMAllowedPrincipals has the **Create database** permission.

   This is done so that we can fully enable Lake Formation and remove any backward compatibility access to Glue Data catalog.

7. Choose **Revoke**

II.1.3 REGISTER AMAZON S3 PATH

With AWS Lake Formation, you register Amazon Simple Storage Service (Amazon S3) paths and add them as storage to a data lake. You can then use Lake Formation permissions for access control to Data Catalog objects that point to these paths, and to the underlying data in the paths. In this step, you register an Amazon Simple Storage Service (Amazon S3) path as the root location of your data lake.

1. Ensure that you are still signed in as the data lake administrator, the **datalake_admin** user.
2. On the Lake Formation console, in the navigation pane, under **Register and ingest**, choose **Data lake locations**.
3. Choose **Register location**, and then choose **Browse**.
4. Select the **S3 bucket** whose name matches the **DataLakeLocation** value that you copied from the Outputs tab the Cloudformation stack. Accept the default role **AWSServiceRoleForLakeFormationDataAccess**, and then choose **Register location**.

II.1.4 GRANT DATA LOCATION PERMISSIONS

IAM Principals must have data location permissions on a data lake location to create Data Catalog tables or databases that point to that location. You must grant data location permissions to the IAM role for Lake Formation blueprint so that workflow can write to the data ingestion destination.

1. Ensure that you are still signed in as the data lake administrator, the **datalake_admin** user.
2. On the Lake Formation console, in the navigation pane, under **Permissions**, choose **Data locations**.
3. Choose **Grant**, and in the **Grant permissions** dialog box, do the following:
    a. For **IAM user and roles**, choose **LakeFormationWorkflowRole**.
    b. For **Storage locations**, choose your S3 bucket that you registered in the previous step.
4. Choose **Grant**.

II.1.5 EDIT DATABASE SETTING IN DATA CATALOG

Metadata tables in the Lake Formation Data Catalog are stored within a database.

1. On the Lake Formation console, in the navigation pane, under **Data catalog**, choose **Databases**.
2. You should be able to see '**amazon-grocery-reviews**' database. This data catalog database was pre-created.
3. Select the amazon-grocery-reviews database and click on **Actions → Database → Edit**.
4. Uncheck the box that says '**Use only IAM access control for new tables in this database**'.

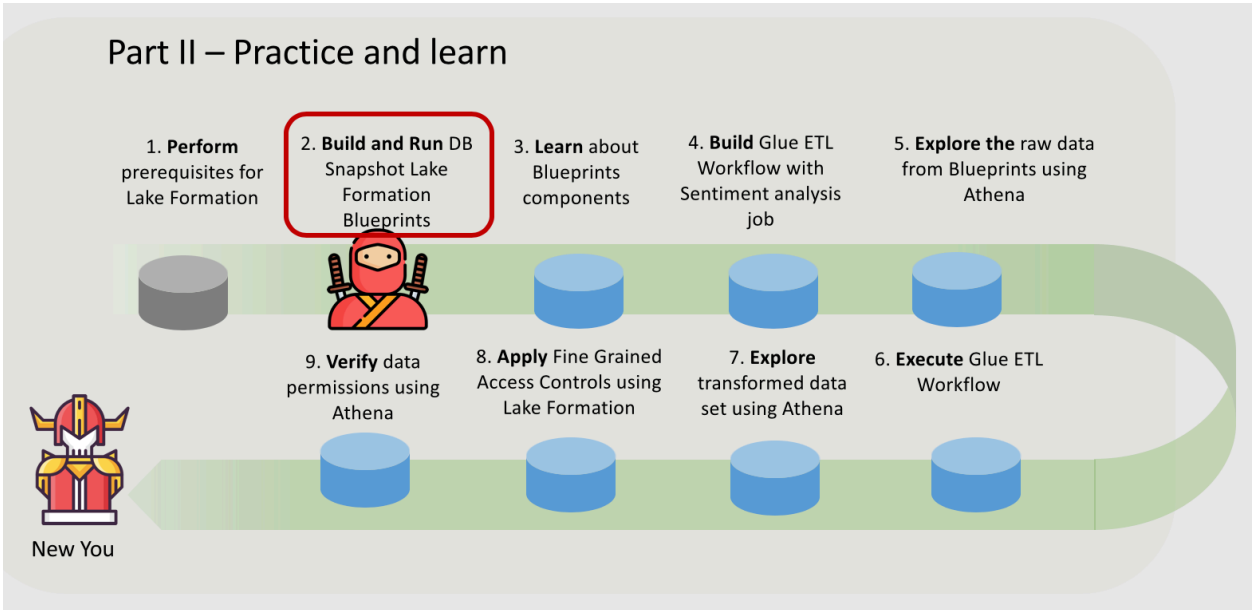
5. Click **Save**.

II.1.6 GRANT DATA PERMISSIONS

You must grant permissions to create metadata tables in the Data Catalog. Because the Lake Formation blueprint runs with the role LakeFormationWorkflowRole, you must grant these permissions to the role.

1. On the Lake Formation console, in the navigation pane, under **Permissions**, choose **Data permissions**.
2. Choose **Grant**, and in the **Grant permissions** dialog box, do the following:
    a. For **IAM user and roles**, choose **LakeFormationWorkflowRole**.
    b. For **Database**, choose the database, **amazon-grocery-reviews**.
    c. For **Database permissions**, make sure **Super** is checked and clear any of the **Grantable permissions** if it is selected.
3. Choose **Grant**.

II.2 BUILD AND RUN DB SNAPSHOT LAKE FORMATION BLUEPRINT



In this step, we will use one of the predefined Lake Formation blueprints to ingest data from the 'amazon_grocery_reviews' table in MySQL RDS into our S3 DataLake. The AWS Lake

Formation blueprint will generate the AWS Glue jobs, crawlers, and triggers that will discover and ingest data into the data lake.

II.2.1 CREATE A DBSNAPSHOT LAKE FORMATION BLUEPRINT

1. On the Lake Formation console, in the navigation pane, choose **Blueprints**, and then choose **Use blueprint**.
2. On the **Use a blueprint** page, under **Blueprint type**, choose **Database snapshot**.
3. Under **Import source**, for **Database connection**, choose the connection that has been pre-created, GlueDBConnection-XXXXXX.
4. For **Source data path**, enter the path from which to ingest data. For JDBC databases with schema support, enter **mysqldb/amazon_grocery_reviews**
5. Under **Import target**, specify these parameters:

Target database	amazon-grocery-reviews
Data lake location	s3://<your_DataLakeLocation>
Data format	parquet

6. For import frequency, choose **Run on demand**.
7. Under **Import options**, specify these parameters:

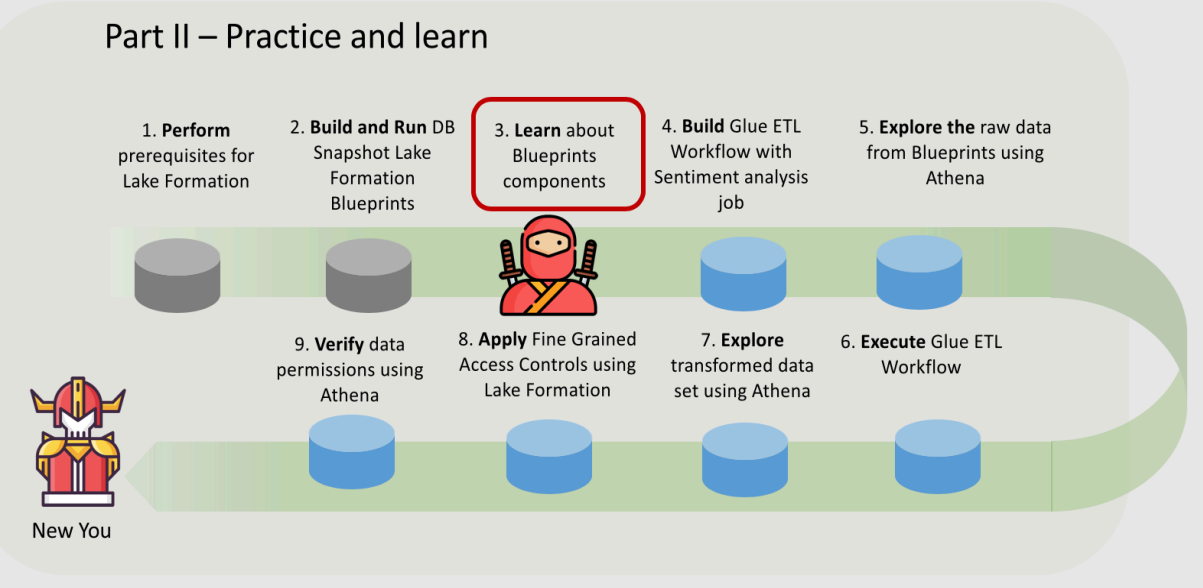
Workflow name	Ingest-workflow
IAM role	LakeFormationWorkflowRole
Table prefix	raw **Note** Must be lower case.
Maximum Capacity	2
Concurrency	1

*** **For the successful execution of this workshop, DO NOT change the table prefix to something else. None of the subsequent Glue ETL jobs in the Glue workflow will work if this value is changed**

8. Choose **Create**, and wait for the console to report that the workflow was successfully created.
9. Once the blueprint gets created, click on **Start it Now?**

   There may be a delay of 5- 10s delay in the blueprint showing up. You may have to hit refresh. Select the blueprint and choose Start in Actions drop down.

II.3 LEARN ABOUT LAKE FORMATION BLUEPRINTS COMPONENTS



II.3.1 EXPLORE THE UNDERLYING COMPONENTS OF A BLUEPRINT

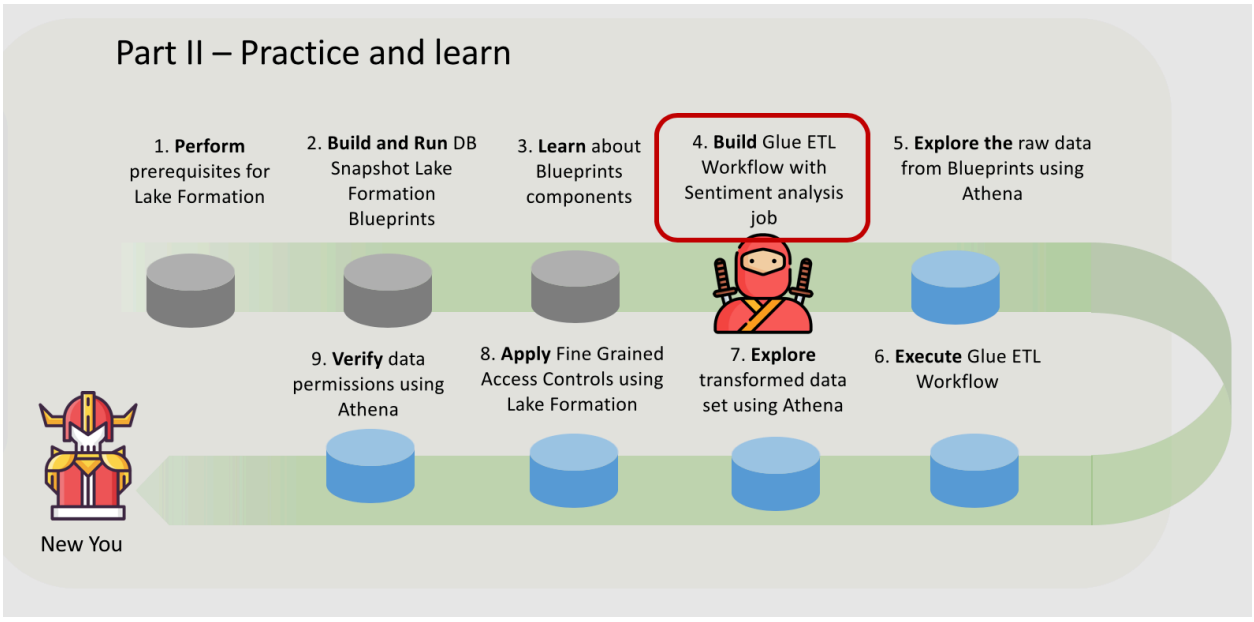
The Lake Formation blueprint creates a Glue Workflow under the hood which contains Glue ETL jobs – both python shell and pyspark; Glue crawlers and triggers. It will take somewhere between 15-20 mins to finish execution. In the meantime, let us drill down to see what it creates for us.

1. On the Lake Formation console, in the navigation pane, choose **Blueprints**
2. In the Workflow section, click on the Workflow **name**. This will direct you to the Workflow run page. Click on the **Run Id.**

3. On the **Glue Console page**, click on **Get Started** and head to **ETL → Workflows** in the navigation pane.
4. Here you can see the graphical representation of the Glue workflow built by Lake Formation blueprint. Highlighting and clicking on individual components will display the details of those components (name, description, job run id, start time, execution time)
5. To understand what all Glue Jobs got created as a part of this workflow, in the navigation pane, click on **Jobs**.
6. All the jobs that start with '**Ingest-Workflow'** were created as a part of the Lake Formation blueprint. The Glue workflow created by the blueprint uses these jobs and ties them together via triggers and crawlers.
7. Every job comes with history, details, script and metrics tab. Review each of these tabs for any of the python shell or pyspark jobs.

**** While the blue print is running, we can proceed to setting up the Glue ETL pipeline (STEP II.4) ****

II.4 BUILD AN AWS GLUE ETL WORKFLOW WITH SENTIMENT ANALYSIS JOB



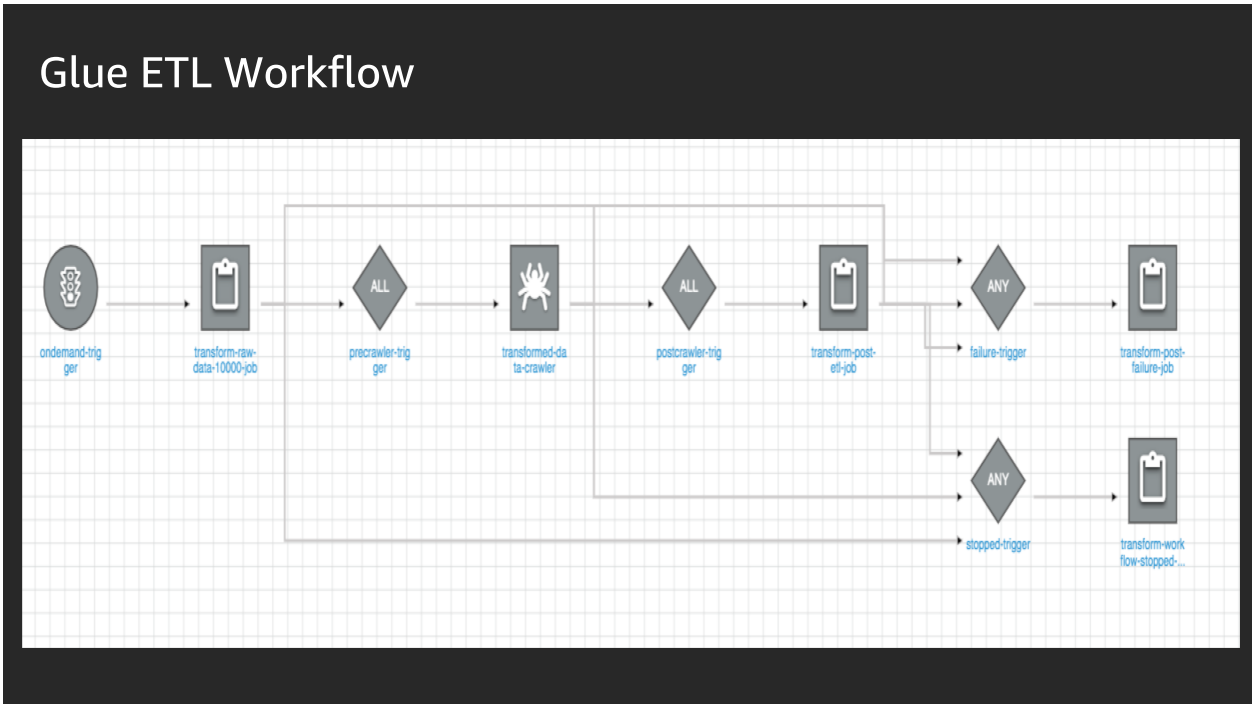
While we wait for the Lake Formation blueprint to finish execution and ingest our raw data into our S3 data lake, let us build out our ETL workflow. The Glue ETL workflow that we will be building out will transform the raw amazon groceries reviews data into a new dataset with review sentiments in it. The transformed data will be parquet format with snappy compression

so that it is optimized for querying. Now, let's go ahead and build our ETL workflow using AWS Glue.

In this section, we'll build an AWS Glue ETL workflow that looks as follows. An on-demand trigger will kick off the **transform-raw-data-1000-job** Glue job(pyspark) which contains the ETL logic of sentiment analysis. This job will append the data set with a new column with the deduced sentiment on the review. For the sake of brevity and speed, we have chosen to run the sentiment analysis on a subset of the grocery reviews dataset. The sentiment analysis job will only process 1000 reviews.

Once the pyspark ETL job finishes and writes the transformed data set to our S3 data lake, a **glue crawler** will be triggered and catalog the **transformed table** in the glue data catalog. If everything runs successfully, the **post-etl python shell** script will be executed to relay the successful completion of the workflow else if any stage fails or is stopped/cancelled, the corresponding python shell script will be triggered and notify you of the outcome.

Here is how our ETL workflow will look like.



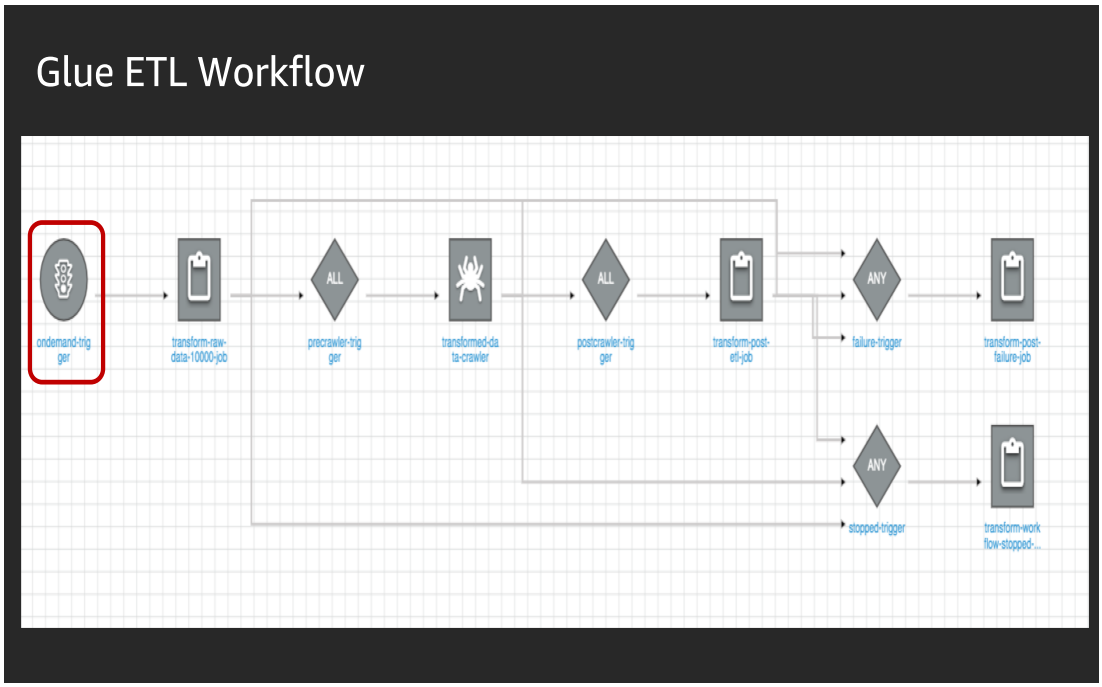
II.4.1 CREATE A GLUE WORKFLOW

Let us start building out our Glue ETL Workflow.

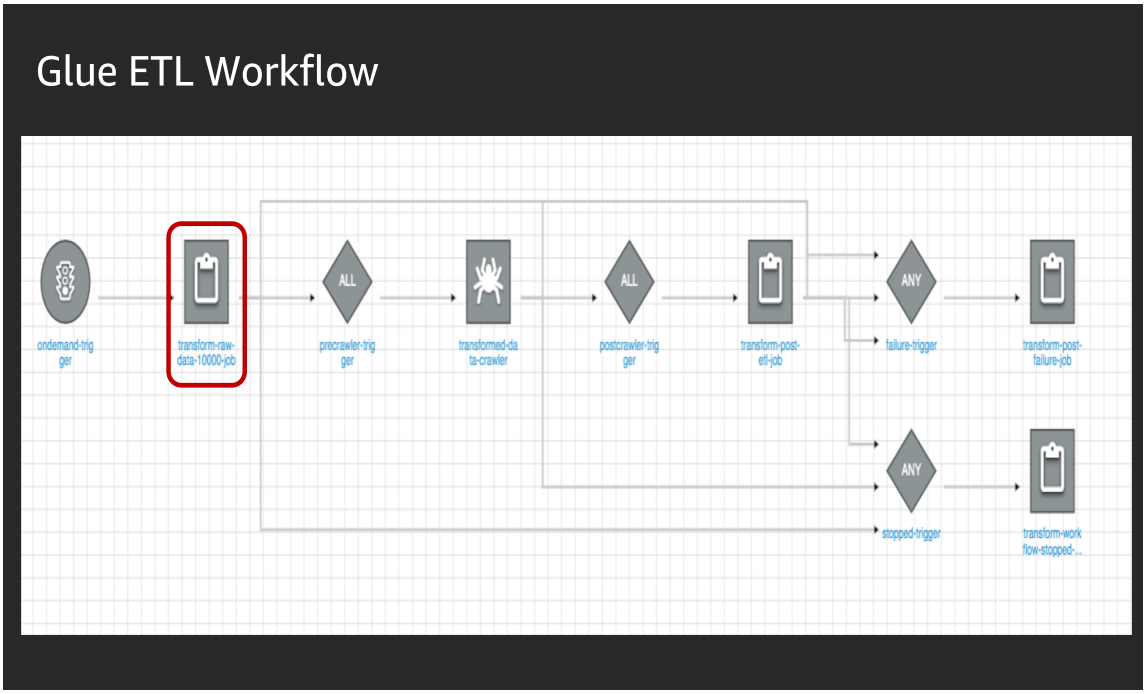
1. Ensure you are logged in as **datalake_admin** user.
2. Navigate to **Glue console**.

3. In the navigation pane, under ETL, select **Workflows**.
4. Click on **Add Workflow** button.
5. Give the workflow a name. Let's call it **Etl-workflow** and click on **Add workflow**.
6. Select the workflow you have just created and in the **Graph** tab, click on **Add Trigger**
7. Select **Add New** tab and put in the following values and then select Add

Name	ondemand-trigger
Trigger Type	OnDemand

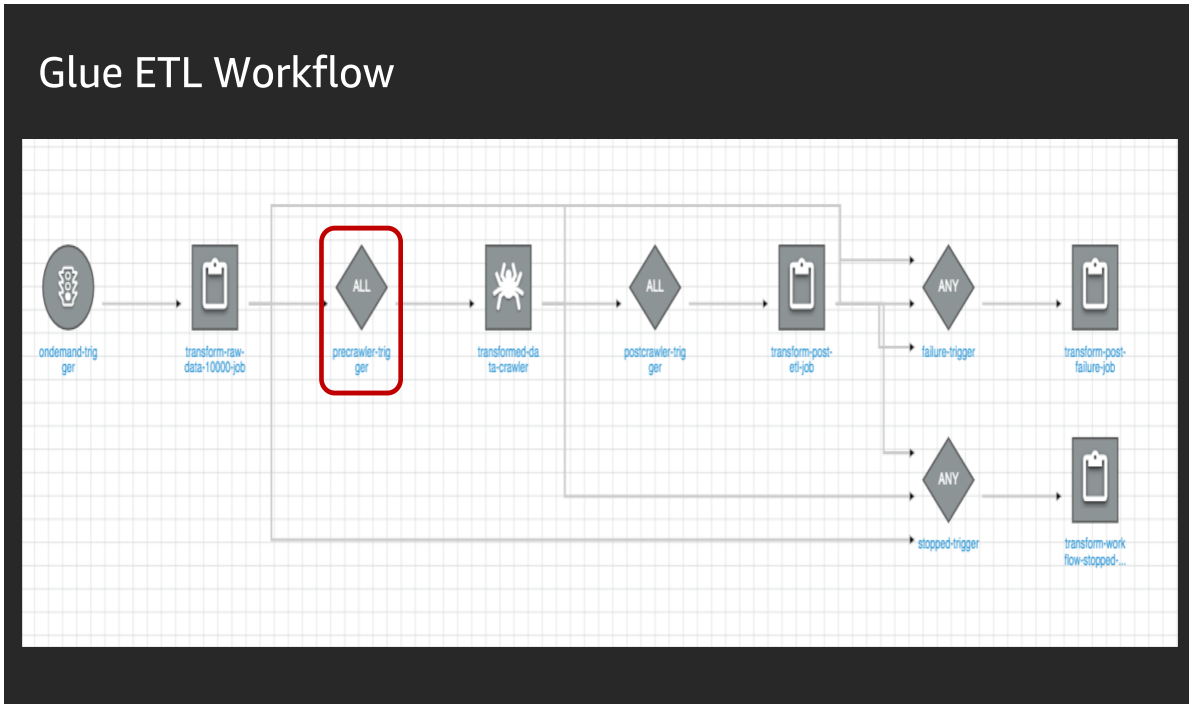


8. On the graph you would now be seeing 2 figures – round depicting the ondemand-trigger and a box with **Add node** message in it. You will also see a link chain icon next to the ondemand-trigger. This is an indication that the trigger is currently incomplete. It will be completed when we add a node to this trigger. Let us do that then. Click on **Add Node** and from the dialog box, in **Jobs** tab, select **transform-raw-data-1000-job** and click **Add**. If you can't find it in the list, put the name of the job in the search bar and find the job. This pyspark job utilizes Amazon Comprehend to perform sentiment analysis on the 1000 reviews data set that we have just ingested in our data lake. We also have the pyspark job that performs sentiment analysis on the entire dataset, called transform-raw-data-all-job. You can try building a Glue workflow with this job as your homework. For this workshop we will consider only the first 1000 reviews and run sentiment analysis on that.

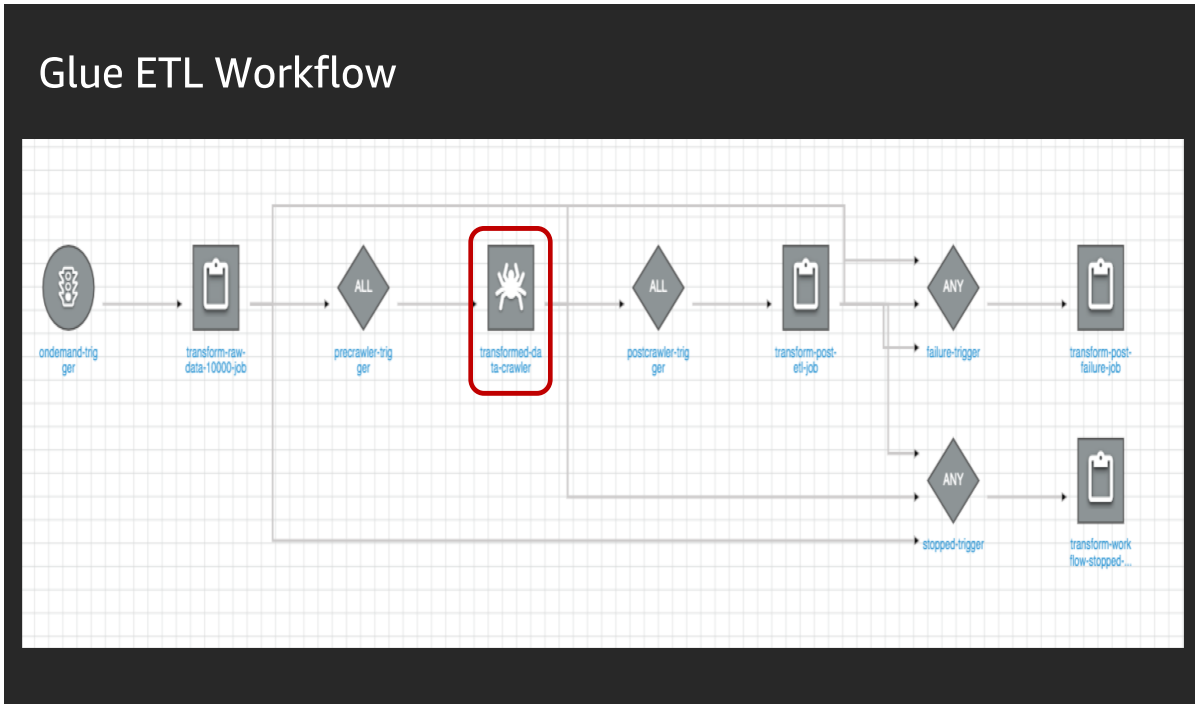


9. Next, highlight the **transform-raw-data-1000-job** box in the graph. You will be directed to adding a trigger. Click on **the Add Trigger** diamond box in the graph and select **Add New** tab. Plug in the following values. Scroll down to find the 'Start after ALL Watched event' and Click Add.

Name	precrawler-trigger
Trigger Type	Event
Trigger Logic	Start after ALL watched event

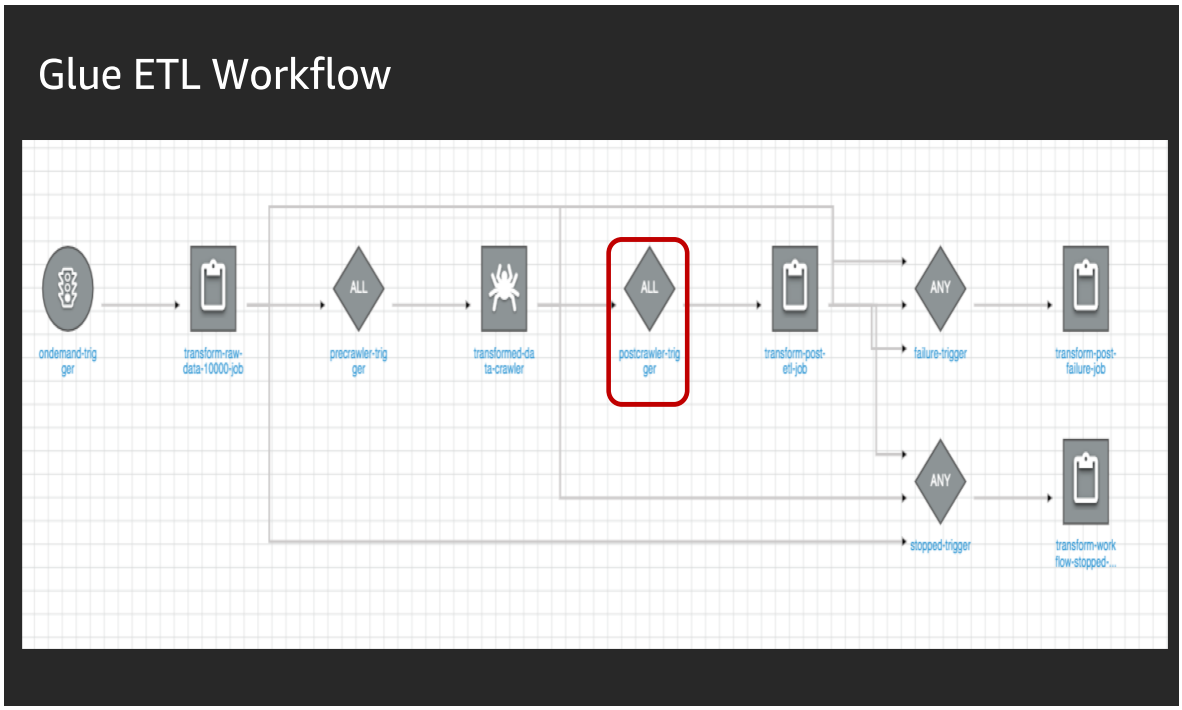


10. Select the **Add Node** box to the right of this newly added **precrawler-trigger (ALL trigger**) and from the **crawlers** tab, select **transformed-data-crawler** and click on Add.

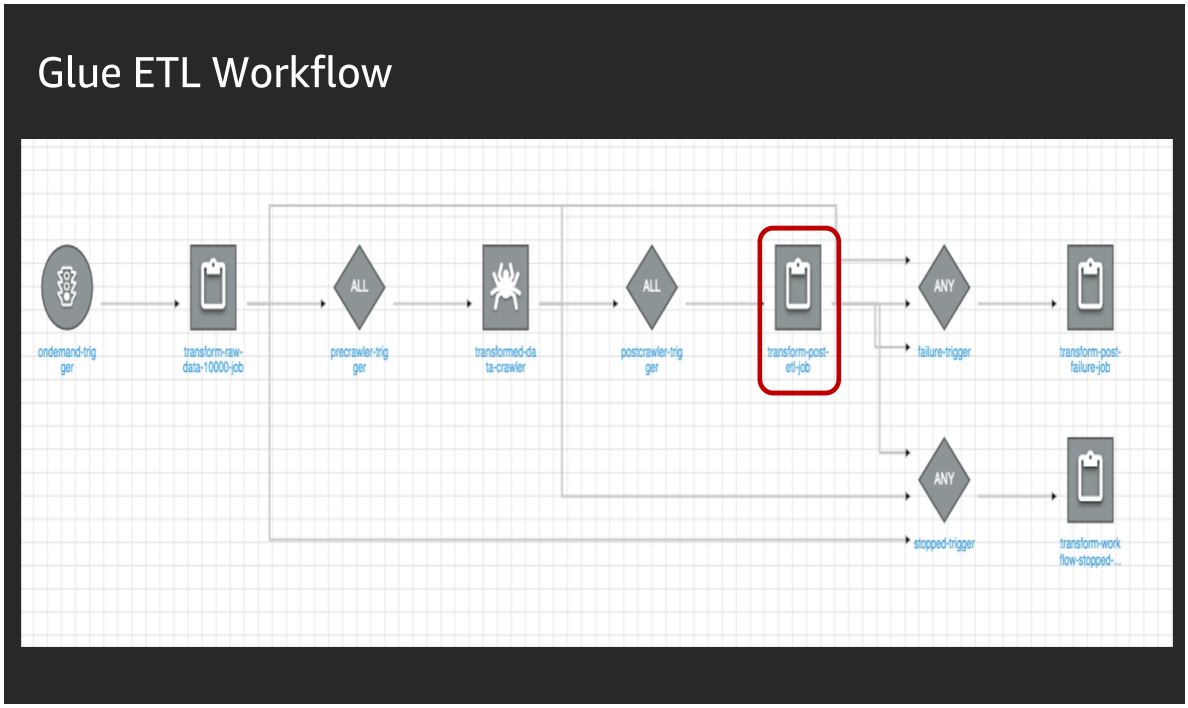


11. Click on the newly added crawler and highlight the **Add trigger** diamond box. Select Add New tab and plug in the following values

Name	postcrawler-trigger
Trigger Type	Event
Trigger Logic	Start after ALL watched event



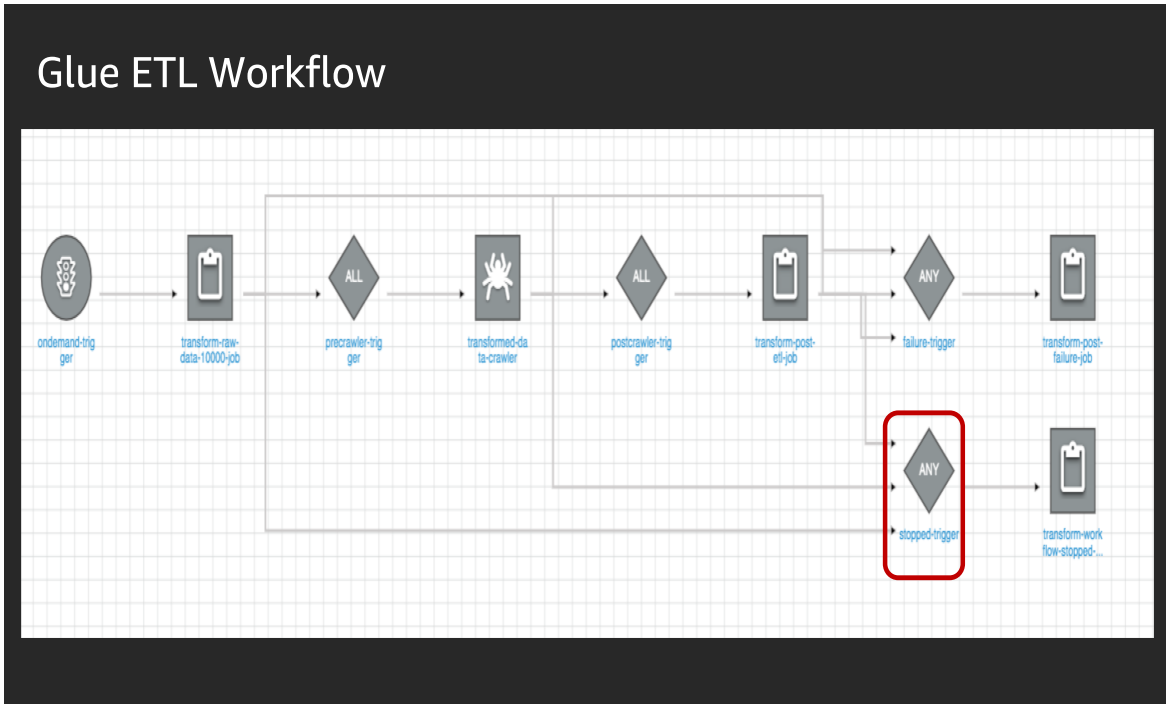
12. Click on **the Add Node** appearing to the right of the **postcrawler trigger (ALL Trigger).** Make sure you are on the Jobs tab. Select **transform-post-etl-job** and click on Add. If you can't find it in the list, put the name of the job in the search bar and find the job.



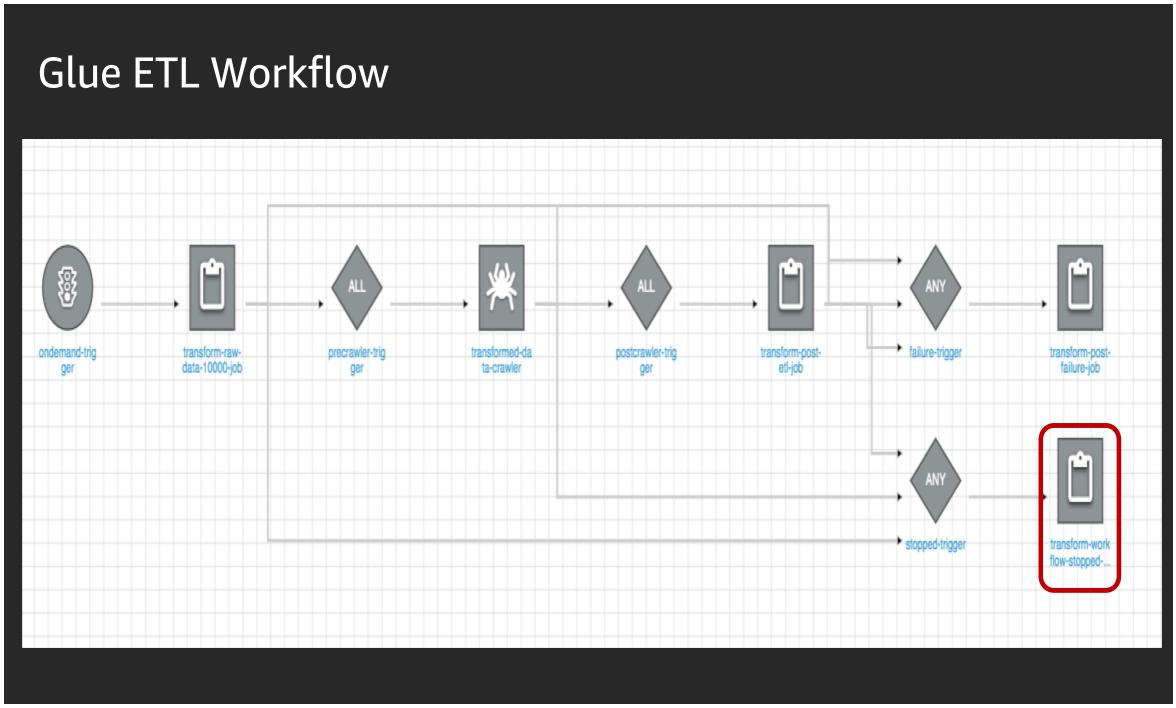
[OPTIONAL STEPS - The following steps will show you how to use ANY trigger. These steps are not mandatory for the successful execution of your workflow. Attempt them if you have time]

13. Next, let's add **ANY** triggers. We will be adding two ANY triggers in this Glue workflow, one which gets triggered when any job/crawler in the workflow is STOPPED and the other which gets triggered if any job/crawler in the workflow FAILS. Let us get started. Highlight the **transform-post-etl-job** and select **Add Trigger** diamond box in the workflow. Plug in the following values.

Name	stopped-trigger
Trigger Type	Event
Trigger Logic	Start after ANY watched event



14. Select the Add Node box to the right of this newly added ANY trigger and from the Jobs tab, select **transform-workflow-stopped-job** and click on **Add.**



15. Next, highlight the **ANY stopped-trigger**, and click on the **Action** button at the top left and select **Add job(s) and crawler(s) to watch**.
16. From the dialog box, in the **Jobs tab**, select **transform-raw-data-1000-job** and choose **job event as STOPPED.** Click Add. A lined arrow should appear between transform-raw-data-1000-job and Stopped-Trigger.
17. Once again, highlight the **ANY stopped-trigger**, and click on the **Action** button at the top left and select **Add job(s) and crawler(s) to watch**.
18. This time, in the dialog box, in the **Crawlers tab** and select **transformed-data-crawler** and **choose job event as CANCELLED**. Click Add. A lined arrow should appear between transform-data-crawler and Stopped-Trigger.
19. Now, click on **transform-post-etl-job** box on the left side of the **stopped-trigger (ANY)** and click on the **Action** button at the top left and **select Edit watched event** and updated with the following values.

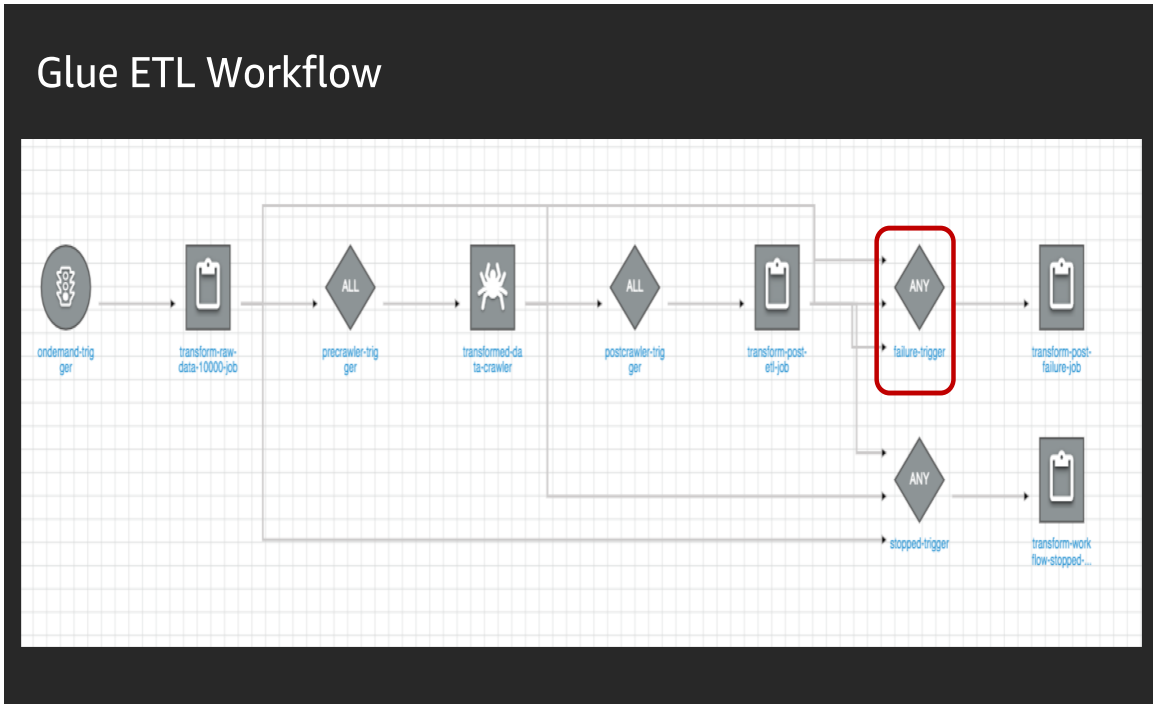
Select the Trigger context	stopped-trigger
Event to watch for	Stopped

20. Click on the **stopped-trigger (ANY)** and just check the watching section. It should have the following values.

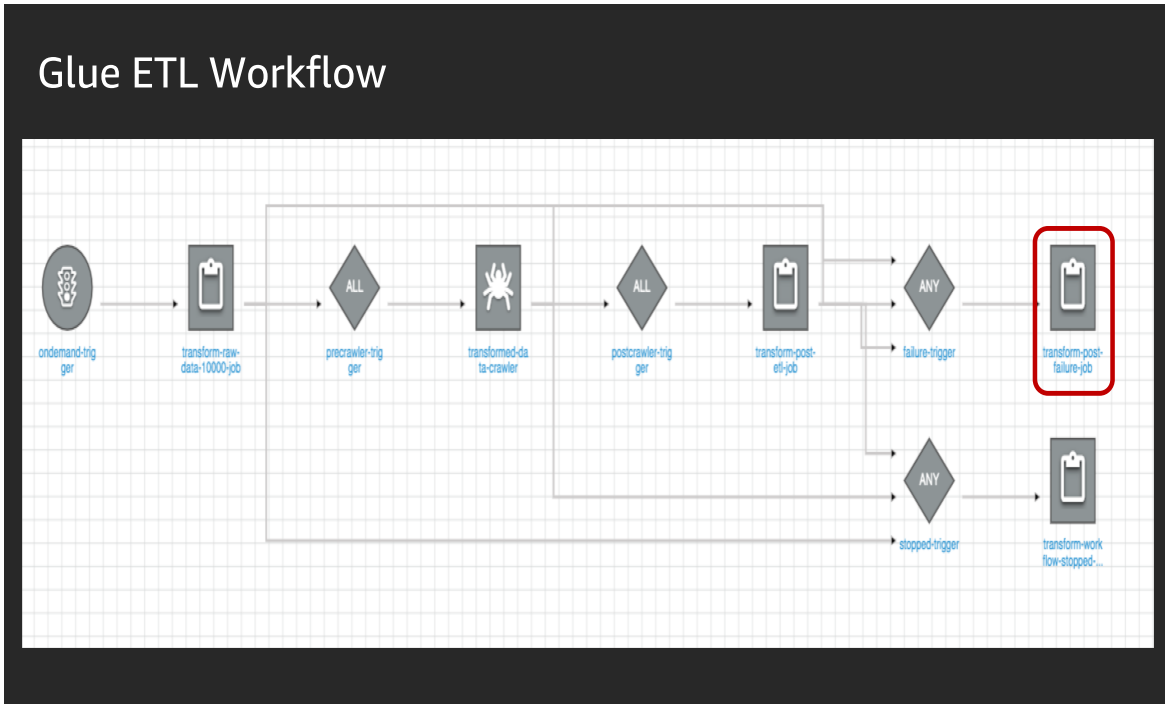
Transform-post-etl-job	STOPPED
Transform-raw-data-1000-job	STOPPED
Transformed-data-crawler	CANCELLED

21. We are now left with the final component of this workflow – which is the second ANY trigger. This would get triggered if any of the previous steps fail. Let's go about adding that. Highlight the **transform-post-etl-job** and select **Add Trigger** diamond box in the workflow. Plug in the following values.

Name	failure-trigger
Trigger Type	Event
Trigger Logic	Start after ANY watched event



22. Select **the Add Node** box to the right of this newly added ANY trigger and from the Jobs tab, select **transform-post-failure-job** and click on Add.



23. Next, highlight the ANY Failure-Trigger, and click on the **Action** button at the top left and select Add job(s) and crawler(s) to watch.

24. From the dialog box, in **the Jobs tab**, select **transform-raw-1000-job** and **choose job event as FAILED**. Click Add. A lined arrow should appear between transform-raw-data-job and Failure-Trigger.
25. Once again, highlight **the ANY Failure-Trigger**, and click on **the Action** button at the top left and select **Add job(s) and crawler(s) to watch**.
26. This time, in the dialog box, in the **Crawlers tab** and select **transformed-data-crawler** and **choose job event as FAILED**. Click Add. A lined arrow should appear between transform-data-crawler and Failure-Trigger.
27. Now, **click on transform-post-etl-job box** on the left side of the **failure-trigger (ANY)** and click on the Action button at the top left and select **Edit watched event.**
28.

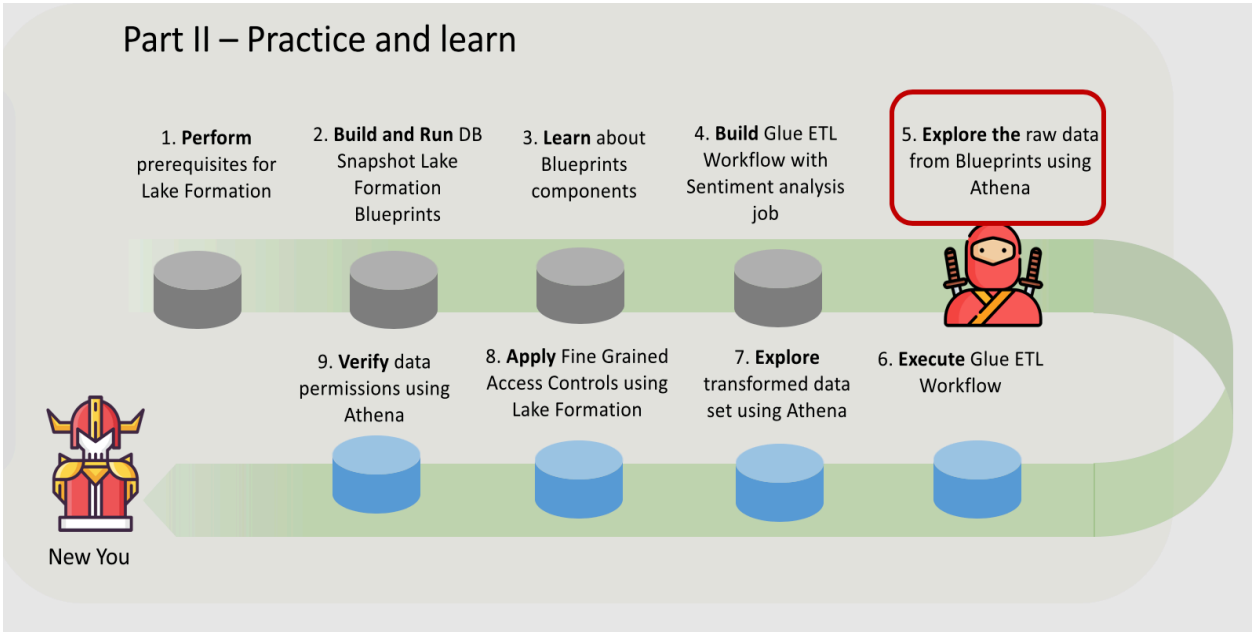
Select the Trigger context	failure-trigger
Event to watch for	Failed

29. Click on the **failure-trigger (ANY)** and just check the watching section. It should have the following values.

Transform-post-etl-job	FAILED
Transform-raw-data-1000-job	FAILED
Transformed-data-crawler	FAILED

**** PROCEED TO STEP II.5 ONLY IF LAKE FORMATION BLUEPRINT HAS SUCCESSFULLY COMPLETED. ****

II.5 EXPLORE THE RAW DATA SET USING AMAZON ATHENA



In step II.2, you built and ran a Lake Formation blueprint that ingested data from MySQL DB into your S3 Data Lake. Let us explore this raw dataset by querying it in Athena.

II.5.1 VALIDATE THE DATA LAKE AFTER LAKE FORMATION WORKFLOW COMPLETES

Once the blueprint completes successfully, let us validate whether the data has indeed made its way to the S3 data lake.

1. On the **S3 console**, select your **DataLakeLocation**. You will see a prefix, namely, **raw_mysqldb_amazon_grocery_reviews** and then **version_0** under raw_mysqldb_amazon_grocery_reviews. Here in this S3 location, you should be able to see the ingested file. This is the data set that moved from the database into S3 in parquet format.
2. Next, go to **Lake Formation console** and click on **Database** in navigation pane. Select **amazon-grocery-reviews** database and click on **View Tables**. You will see the **raw_mysqldb_amazon_grocery_reviews** table.
3. Select the **raw_mysqldb_amazon_grocery_reviews** table. On the details page, you will see that this catalog table has parquet classification and points to the same S3 data lake

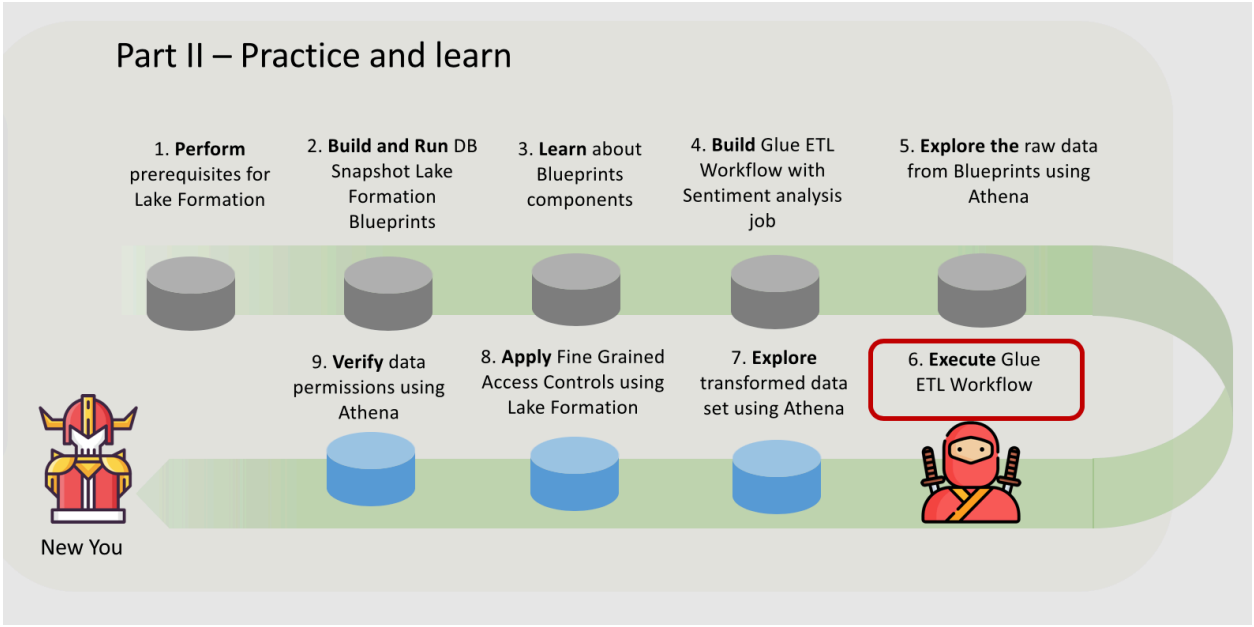
location as mentioned above. Scroll down on this page and check the schema structure of the catalog table

II.5.2 QUERY RAW DATA WITH AMAZON ATHENA

Now that the data has been cataloged by Lake Formation blueprints, let's run a set of sample data exploration queries on the **Amazon grocery reviews dataset**.

1. Navigate to **Amazon Athena** console
2. Before you run your first Athena query, you need to **set up a query result location** in Amazon S3. On the right side of the Athena console, click on **Settings** and type in the S3 bucket location. This should be the value of **AthenaQueryResultsBucket** from the **Output tabs of your Cloudformation**. The entry should look something like this: s3:// athenaqueryresults-<Accountnumber>/ and **hit Save.**
3. Ensure database **amazon-grocery-reviews** is selected.
4. Click on the tab **Saved Queries**
5. In the search field, type **Explore_Raw_Data_Query**
6. Double click on the query **Explore_Raw_Data_Query** to open in Athena query editor
7. This saved query has four queries in it. Highlight each query one by one and click **Run query**

II.6 EXECUTE THE GLUE WORKFLOW



In step 11.4, you built the Glue ETL workflow. Let us go ahead and execute it. This workflow should take 10-15 mins to complete.

II.6.1 GRANT DATA PERMISSIONS ON TABLE TO LAKEFORMATIONWORKFLOWROLE

Before we start the execution of our Glue Workflow, let us grant table level permissions to LakeFormationWorkflowRole. During our prerequisites, we granted Super permissions to LakeFormationWorkflowRole on the Database: amazon_grocery_reviews but after the Lake Formation Blueprint ran, it created the raw_mysqldb_amazon_grocery_reviews table in this database and the Glue Workflow performs ETL on this table. Therefore, we need to grant the LakeFormationWorkflowRole which will be executing the sentiment analysis ETL job, with the appropriate permissions to be able to access this table.

1. On the Lake Formation console, in the navigation pane, under **Permissions**, choose **Data permissions**.
2. Choose **Grant**, and in the **Grant permissions** dialog box, do the following:
    a. For **IAM user and roles**, choose **LakeFormationWorkflowRole**.
    b. For **Database**, choose the database, **amazon-grocery-reviews**.
    c. For **Table**, choose the table, **raw_mysqldb_amazon_grocery_reviews**.
    d. For **Table permissions**, choose **Super**, and clear any of the **Grantable permissions** if it is selected.

II.6.2 EXECUTE THE GLUE WORKFLOW

Finally, we have everything in place to kickstart our Glue ETL workflow.

1. Navigate to the AWS Glue console
2. In the left menu, under **ETL**, click **Workflows**, then select the **Etl-workflow** we created in the previous step.
3. Under **Actions**, select **Run**.
4. To check the status of the workflow, select it and go to **Histor**y tab. This would have the workflow id listed in it. Select the workflow id and click on **View Run Details**. This will direct you to the execution graph of the workflow. The stage that has successfully executed will appear green and the component that is currently in progress will have a spinning work in progress icon in its upper right corner.
5. The workflow should take somewhere between 10-15 mins to complete. Once, the workflow successfully completes, proceed to the next step.

II.6.3 VALIDATE THE OUTPUT OF THE GLUE ETL WORKFLOW

Now that our Glue ETL workflow has successfully completed, let us validate the output.

1. Navigate to AWS S3 and select your DataLakeLocation. Under this bucket, you should now see a transformed S3 prefix. This subfolder gets created as a part of the transform-raw-data-job.
2. In this transformed subfolder, you will notice one more subfolder, this time partitioned by the SENTIMENT. You will have 4 sentiment subfolders, namely, **Sentiment=Positive, Sentiment=Negative, Sentiment=Mixed and Sentiment=Neutral**. Each partitioned bucket contains parquet files with snappy compression. Our glue job – **transform-raw-data-1000-job** performed sentiment analysis on the raw data using Amazon Comprehend and put the dataset into partitioned S3 buckets with parquet and snappy compression.
3. Next, navigate to **Lake Formation console**. In the navigation pane, select Database, and select **amazon-grocery-reviews**. Select **View Tables** on the top right. You will see a new table names**, transformed** appear in this database. This table was crawled by the **transformed-data-crawler** component of our Glue Workflow.

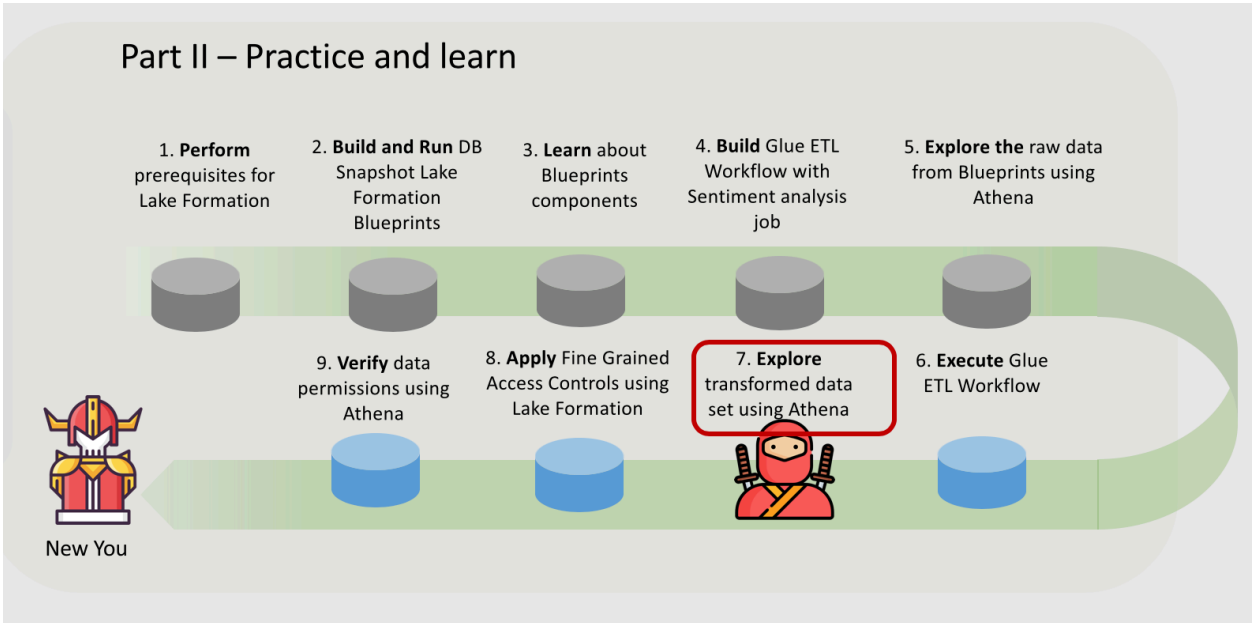
II.6.4 GRANT DATA PERMISSIONS ON TABLE TO DATALAKE_ADMIN

Before we start the querying the data in transformed table, let us grant table level permissions to datalake_admin user.

1. On the Lake Formation console, in the navigation pane, under **Permissions**, choose **Data permissions**.
2. Choose **Grant**, and in the **Grant permissions** dialog box, do the following:
    a. For **IAM user and roles**, choose **datalake_admin**.
    b. For **Database**, choose the database, **amazon-grocery-reviews**.
    c. For **Table**, choose the table, **transformed**.
    d. For **Table permissions**, choose **Select**, and clear any of the **Grantable permissions** if it is selected.

**** PROCEED TO STEP II.7 ONLY IF GLUE ETL WORKFLOW HAS SUCCESSFULLY COMPLETED ****

II.7 EXPLORE TRANSFORMED DATA SET USING AMAZON ATHENA



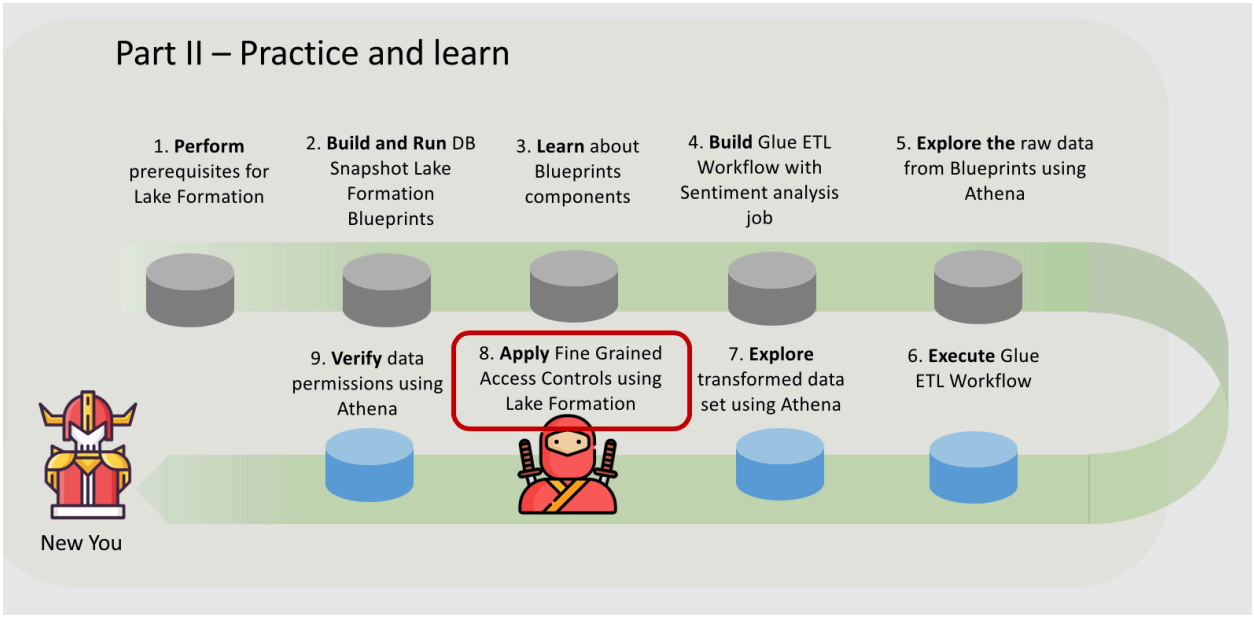
After the Glue ETL workflow successfully completes, you will be able see **transformed** table in the **amazon-grocery-reviews** database in the Glue Catalog. Let us explore this transformed dataset by querying it in Athena.

II.7.1: QUERY TRANSFORMED DATA WITH AMAZON ATHENA

Now that the data has been cataloged by the Glue ETL workflow, let's run a set of sample data exploration queries on the **Transformed amazon grocery reviews dataset**.

1. Navigate to **Amazon Athena** console
2. Ensure database **amazon-grocery-reviews** is selected.
3. Click on the tab **Saved Queries**
4. In the search field, type **Explore_Transformed_Limited_Data_Query**
5. Click on the query **Explore_Transformed_Limited_Data_Query** to open in Athena query editor
6. This saved query has four queries in it. Highlight each query one by one and click **Run query**

II.8 GRANT FINE GRAIN ACCESS CONTROL TO DATA ANALYST USER



Let us now grant permissions on the new Data Catalog table 'transformed' in AWS Lake Formation so that the data analyst can query the data that the tables point to.

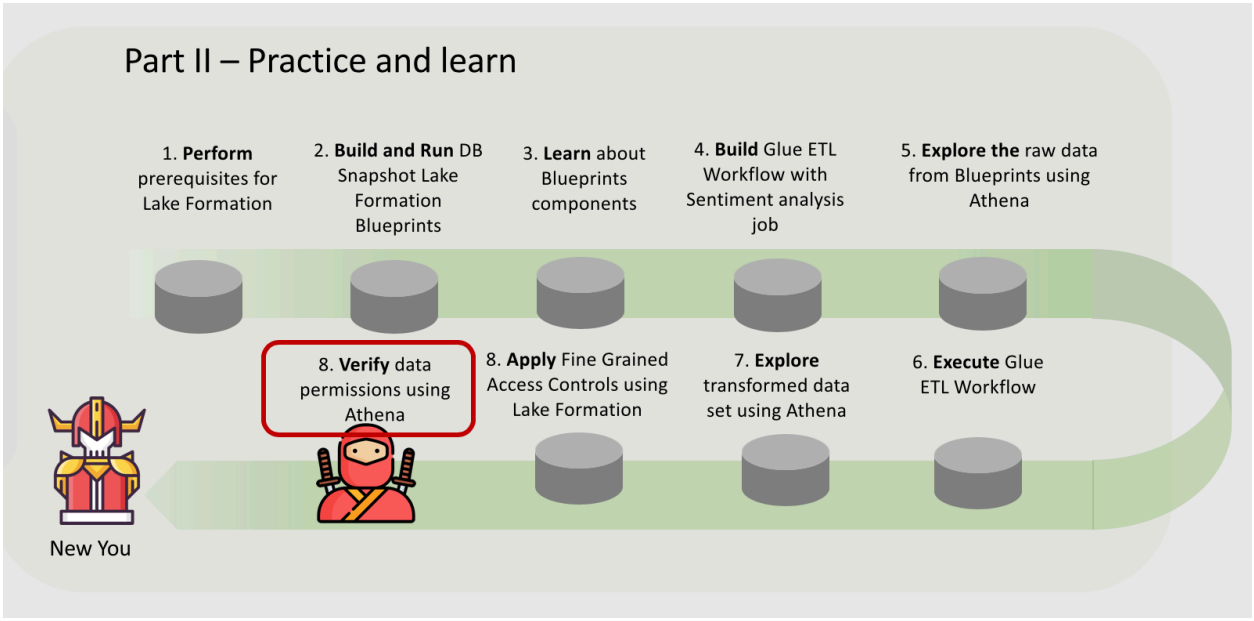
II.8.1 GRANT DATA PERMISSIONS ON TABLE TO DATALAKE_USER

Before we start the querying the data in transformed table, let us grant table level permissions to **datalake_user.**

1. On the Lake Formation console, in the navigation pane, under **Permissions**, choose **Data permissions**.
2. Choose **Grant**, and in the **Grant permissions** dialog box, do the following:
    a. For **IAM user and roles**, choose **datalake_user.**
    b. For **Database**, choose **amazon-grocery-reviews**.
    c. The **Table** list populates.
    d. For **Table**, choose table **transformed.**
    e. For **Columns**, select **Include Columns** and choose **review_id, product_id, product_title, total_votes**
    f. For **Table permissions**, choose **Select**.
3. Choose **Grant**.

**** MAKE SURE YOU LOG OUT FROM datalake_admin and LOGIN AS datalake_user USER ****

II.9 VERIFY DATA PERMISSIONS USING AMAZON ATHENA



After the Glue ETL workflow successfully completes, you will be able see **transformed** table in the **amazon-grocery-reviews** database in the Glue Catalog. Let us explore this transformed dataset by querying it in Athena.

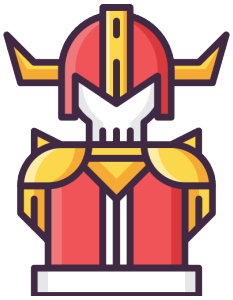
II.9.1: QUERY TRANSFORMED DATA WITH AMAZON ATHENA

Using Athena, let us now explore the transformed data set as **the analyst user**.

1. Sign in as the data analyst user – **datalake_user** (username: **datalake_user** password: **Welcome1**)
2. Navigate to **Amazon Athena** console.
3. Before you run your first Athena query, you need to **set up a query result location** in Amazon S3. On the right side of the Athena console, click on **Settings** and type in the S3 bucket location. This should be the value of **AthenaQueryResultsBucket** from the **Output tabs of your Cloudformation**. The entry should look something like this: s3:// athenaqueryresults-<Accountnumber>/
4. Next, ensure database **amazon-grocery-reviews** is selected.
5. In the navigation pane, you will see that the **datalake_user** can only see transformed table. The datalake_user cannot see **raw_mysql_amazon_grocery_reviews.** This is because, datalake_admin gave **datalake_user** permissions to only select from **transformed** table.

6. Also, drill down into the table and you will see that datalake_user can only see four columns on this table. He has no access to the remainder columns of the transformed table.
7. In the pop-up menu beside the transformed tables, choose **Preview table.**
   The query runs and displays 10 rows of data.

You've made it!



For any questions or comments, please reach out to ant201-workshop@amazon.com